package total.store.dao.impl;  
  
import total.store.dao.RecoveryOrderDao;  
import total.store.model.Order;  
import total.store.model.RecoveryOrder;  
import total.store.util.DbUtil;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
  
public class RecoveryOrderDaoImpl implements RecoveryOrderDao {  
 @Override  
 public Boolean recoveryOrder\_add(RecoveryOrder recoveryOrder) {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con=null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
 String dateNow = sdf.format(new Date());  
 StringBuilder sb = new StringBuilder();  
 sb.append("insert into 置换信息表 values(?,?,?,?,?,'待发货')");  
 ps = con.prepareStatement(sb.toString());  
 ps.setString(1,recoveryOrder.getRecoveryOrderid());  
 ps.setString(2,recoveryOrder.getRecoverybookid());  
 ps.setInt(3,recoveryOrder.getRecoveryAddressid());  
 ps.setDouble(4,recoveryOrder.getRecoveryOrderprice());  
 ps.setString(5,dateNow);  
 int num=ps.executeUpdate();  
 if(num==1){  
 return true;  
 }else {  
 return false;  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public List<RecoveryOrder> recoveryorderlist(int userid) {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("select \* from 置换信息表,用户表,书籍回收表,地址表 where 置换信息表.订单状态 = '待发货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号 and 用户表.编号=?");  
 ps = con.prepareStatement(sb.toString());  
 ps.setInt(1,userid);  
 rs=ps.executeQuery();  
 List<RecoveryOrder> recoveryorderlist=new ArrayList<>();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("置换信息表.编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 StringBuilder sb1 = new StringBuilder();  
 sb1.append("select \* from 置换信息表,用户表,书籍回收表,地址表 where 置换信息表.订单状态 = '待收货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号 and 用户表.编号=?");  
 ps = con.prepareStatement(sb1.toString());  
 ps.setInt(1,userid);  
 rs=ps.executeQuery();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 return recoveryorderlist;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public List<RecoveryOrder> recoveryorderlist\_history(int userid) {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("select \* from 置换信息表,用户表,书籍回收表,地址表 where 置换信息表.订单状态 = '已收货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号 and 用户表.编号=?");  
 ps = con.prepareStatement(sb.toString());  
 ps.setInt(1,userid);  
 rs=ps.executeQuery();  
 List<RecoveryOrder> recoveryorderlist=new ArrayList<>();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("置换信息表.编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 return recoveryorderlist;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public List<RecoveryOrder> recoveryorderlist\_admin() {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("select \* from 置换信息表,书籍回收表,地址表,用户表 where 置换信息表.订单状态 = '待发货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号");  
 ps = con.prepareStatement(sb.toString());  
 rs=ps.executeQuery();  
 List<RecoveryOrder> recoveryorderlist=new ArrayList<>();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 return recoveryorderlist;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public Boolean recoveryOrder\_mo(String recoveryOrderid) {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("update 置换信息表 set 订单状态='待收货' where 编号=?");  
 ps = con.prepareStatement(sb.toString());  
 ps.setString(1,recoveryOrderid);  
 int num = ps.executeUpdate();  
 if (num==1){  
 return true;  
 }else {  
 return false;  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public Boolean recoveryOrder\_user\_mo(String recoveryOrderid) {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("update 置换信息表 set 订单状态='已收货' where 编号=?");  
 ps = con.prepareStatement(sb.toString());  
 ps.setString(1,recoveryOrderid);  
 int num = ps.executeUpdate();  
 if (num==1){  
 return true;  
 }else {  
 return false;  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 public List<RecoveryOrder> recoveryorderlist\_hiadmin() {  
 DbUtil dbUtil =new DbUtil();  
 ResultSet rs = null;  
 PreparedStatement ps = null;  
 Connection con = null;  
 try {  
 con = dbUtil.getCon();  
 if (con == null) {  
 return null;  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("select \* from 置换信息表,书籍回收表,地址表,用户表 where 置换信息表.订单状态 = '待收货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号");  
 ps = con.prepareStatement(sb.toString());  
 rs=ps.executeQuery();  
 List<RecoveryOrder> recoveryorderlist=new ArrayList<>();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 StringBuilder sb1 = new StringBuilder();  
 sb1.append("select \* from 置换信息表,书籍回收表,地址表,用户表 where 置换信息表.订单状态 = '已收货' and 书籍回收表.编号=置换信息表.书籍信息 and 置换信息表.地址信息=地址表.编号 and 地址表.用户信息=用户表.编号");  
 ps = con.prepareStatement(sb1.toString());  
 rs=ps.executeQuery();  
 while(rs.next()) {  
 RecoveryOrder recoveryOrder = new RecoveryOrder();  
 recoveryOrder.setRecoveryOrderid(rs.getString("编号"));  
 recoveryOrder.setRecoverybookname(rs.getString("书籍名"));  
 recoveryOrder.setRecoveryOrderprice(rs.getDouble("订单积分"));  
 Date date=rs.getDate("订单时间");  
 recoveryOrder.setRecoveryOrderdate(date);  
 recoveryOrder.setRecoveryOrderstatus(rs.getString("订单状态"));  
 recoveryOrder.setRecoverycon(rs.getString("收货人"));  
 recoveryOrder.setRecoveryphone(rs.getString("电话"));  
 recoveryOrder.setRecoveryaddrdetail(rs.getString("详细地址"));  
 recoveryorderlist.add(recoveryOrder);  
 }  
 return recoveryorderlist;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }finally {  
 try {  
 dbUtil.closeResultSet(rs);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closeCon(con);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 try {  
 dbUtil.closePs(ps);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
}